Introduction to Modern Cryptography

Michele Ciampi

(Slides courtesy of Prof. Jonathan Katz)

Lecture 7, Part 2

Pseudorandom functions (PRF)
Pseudorandom permutations (PRP)

Random Function

- When we talk about a random function $\boldsymbol{f}$, we mean
  (A) Choosing $\boldsymbol{f}$ uniformly at random (and then fixing it) **or**
  (B) Interacting with $\boldsymbol{f}$
- In particular, once we choose $\boldsymbol{f}$ there is no more randomness involved
- i.e. if we query $\boldsymbol{f}$ on the same input twice, we get the same result

Choosing a Uniform Function

x	$f(x)$
000	010
001	100
010	100
011	111
100	001
101	010
110	010
111	000

- $\mathcal{F}_n$ = all functions mapping $\{0,1\}^n$ to $\{0,1\}^n$
- How big is $\mathcal{F}_n$?
  - Can represent a function in $\mathcal{F}_n$ using $n2^n$ bits
  - $\implies |\mathcal{F}_n| = 2^{n2^n}$
- $n = 3 \implies$ # of entries: $2^3 = 8$

Exercise

How many functions mapping $\{0,1\}^n$ to $\{0,1\}^m$?

1. $m2^n$
2. 2^{n2^m}
3. $m2^{n2^n}$
4. 2^{m2^n}

Exercise

How many functions mapping $\{0,1\}^n$ to $\{0,1\}^m$?

1. $m2^n$
2. 2^{n2^m}
3. $m2^{n2^n}$
4. 2^{m2^n} $\leftarrow$

Choosing a Uniform Function

Method A

Choose uniform $\boldsymbol{f} \in \boldsymbol{\mathcal{F}_n}$

Method B

- For each $\boldsymbol{x} \in \{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{n}}$, choose $\boldsymbol{f}(\boldsymbol{x})$ uniformly in $\{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{n}}$
- i.e. **fill up the function table with uniform values**
- Can view this as being done *on-the-fly*, as values are needed

Pseudorandom Functions (PRF)

- PRF generalizes the notion of PRG
- Instead of **random-looking strings** we have **random-looking functions**

Pseudorandom Functions (PRF)

Informal

A pseudorandom function **looks like** a random (i.e. uniform) function

- As for PRGs, makes no sense to talk about any fixed function being pseudorandom
- We look instead at functions chosen according to some **distribution**
- In particular, we look instead at **keyed functions**

Keyed Functions

Keyed function $\boldsymbol{F_k}$

- Let $\boldsymbol{F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*}$ be an efficient, deterministic algorithm
- Define $\boldsymbol{F_k(x) = F(k, x)}$
- The first input $\boldsymbol{k}$ is called **the key**
- $\boldsymbol{F}$ is efficient $\implies$ $\boldsymbol{F}$ can be computed in poly time **given inputs $\boldsymbol{k}$ and $\boldsymbol{x}$**

Length-preserving Keyed Functions

Length-preserving keyed function F_k

The function F_k is **length-preserving** if:

- $F(k, x)$ only defined if $|k| = |x|$
- and $|F(k, x)| = |k| = |x|$
- i.e. input/s and output of equal size

Uniform Keyed Functions

Choosing a uniform F_k

Choosing a uniform $k \in \{0,1\}^n$ is equivalent to choosing the function $F_k : \{0,1\}^n \to \{0,1\}^n$

F_k induces a distribution on $\mathcal{F}_n$

- **F_k naturally induces a distribution on functions from $\mathcal{F}_n$**
- For uniform $k \in \{0,1\}^n$, $\forall f \in \mathcal{F}_n$:

$$\mathbf{Pr}[f] = \begin{cases} 2^{-n} & f \in \{F_k\} \\ 0 & \text{otherwise} \end{cases}$$

Note

- The number of functions in $\mathcal{F}_n$ is 2^{n2^n}
- $\{F_k\}_{k \in \{0,1\}^n}$ is a subset of $\mathcal{F}_n$
- The number of functions in $\{F_k\}_{k \in \{0,1\}^n}$ is at most 2^n
- **$\{F_k\}$ contains only a tiny fraction of $\mathcal{F}_n$:**

$$\mathbf{2^n \ll 2^{n2^n}}$$

Pseudorandom Functions

Definition

$\boldsymbol{F}$ is a pseudorandom function if $\boldsymbol{F_k}$, for uniform key $\boldsymbol{k} \in \{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{n}}$, is indistinguishable from a uniform function $\boldsymbol{f} \in \boldsymbol{\mathcal{F}_n}$

Pseudorandom Functions

PRG

$\boldsymbol{D}$ is given access to a bit-string

$$\left|\mathbf{Pr}_{x \leftarrow U_n}[D(G(x)) = 1] - \mathbf{Pr}_{y \leftarrow U_{p(n)}}[D(y) = 1]\right| \leq \epsilon(n)$$

Pseudorandom Functions

PRG

$\boldsymbol{D}$ is given access to a bit-string

$$|\mathbf{Pr}_{x \leftarrow U_n}[D(G(x)) = 1] - \mathbf{Pr}_{y \leftarrow U_{p(n)}}[D(y) = 1]| \leq \epsilon(n)$$

PRF

$\boldsymbol{D}$ is given the description of $\boldsymbol{f}$ or $\boldsymbol{F_k}$

$$|\mathbf{Pr}_{k \leftarrow \{0,1\}^n}[D^{F_k(\cdot)} = 1] - \mathbf{Pr}_{f \leftarrow \mathcal{F}_n}[D^{f(\cdot)} = 1]| \leq \epsilon(n)$$

Pseudorandom Functions

PRG

$\boldsymbol{D}$ is given access to a bit-string

$$\left|\mathbf{Pr}_{x \leftarrow U_n}[D(G(x)) = 1] - \mathbf{Pr}_{y \leftarrow U_{p(n)}}[D(y) = 1]\right| \leq \epsilon(n)$$

PRF

$\boldsymbol{D}$ is given the description of $\boldsymbol{f}$ or $\boldsymbol{F_k}$

$$\left|\mathbf{Pr}_{k \leftarrow \{0,1\}^n}[D^{F_k(\cdot)} = 1] - \mathbf{Pr}_{f \leftarrow \mathcal{F}_n}[D^{f(\cdot)} = 1]\right| \leq \epsilon(n)$$

Problem

- Description of $\boldsymbol{f}$ is at least $\boldsymbol{n2^n}$ bits long i.e. exponential
- $\boldsymbol{D}$ has polynomial capabilities

Pseudorandom Functions

PRF

$\boldsymbol{D}$ is given ~~the description of~~ **oracle access** to $\boldsymbol{f}$ or $\boldsymbol{F_k}$

$$\boldsymbol{|\mathbf{Pr}_{k \leftarrow \{0,1\}^n}[D^{F_k(\cdot)} = 1] - \mathbf{Pr}_{f \leftarrow \mathcal{F}_n}[D^{f(\cdot)} = 1]| \leq \epsilon(n)}$$

Solution

- Now $\boldsymbol{D}$ can query $\boldsymbol{f}$ (resp. $\boldsymbol{F_k}$) at most poly times

Pseudorandom Functions (PRFs)

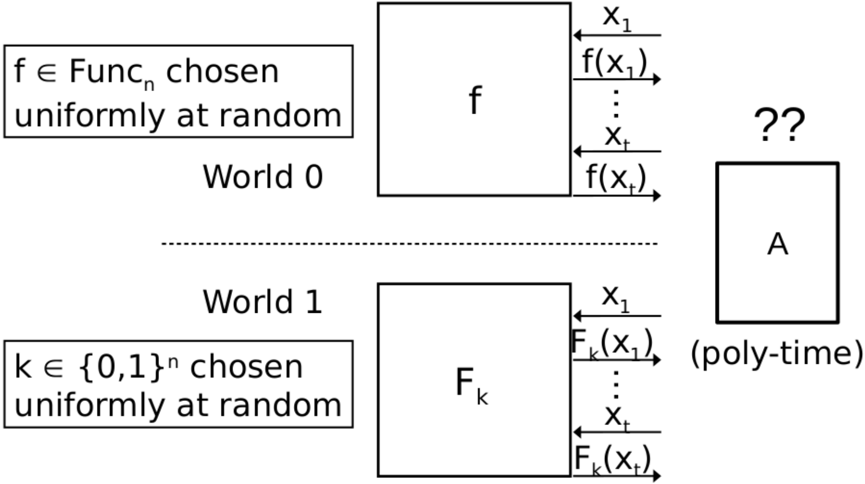
Definition (refined)

$\boldsymbol{F}$ is a pseudorandom function if $\boldsymbol{F_k}$, for uniform key $\boldsymbol{k \in \{0,1\}^n}$, is such that for all **poly-time distinguishers $\boldsymbol{D}$**:

$$\left|\mathbf{Pr}_{k \leftarrow \{0,1\}^n}[D^{F_k(\cdot)} = 1] - \mathbf{Pr}_{f \leftarrow \mathcal{F}_n}[D^{f(\cdot)} = 1]\right| \leq \epsilon(n)$$

$\boldsymbol{D}$ can query $\boldsymbol{f}$ (resp. $\boldsymbol{F_k}$) on any input $\boldsymbol{x}$ **at most poly times**

PRF vs. RF


f ∈ Func$_n$ chosen uniformly at random
World 0
f
x_1
$f(x_1)$
x_t
$f(x_t)$
World 1
k ∈ {0,1}^{n} chosen uniformly at random
F_k
x_1
$F_k(x_1)$
x_t
$F_k(x_t)$
??
A
(poly-time)

PRF vs. RF

Warning

Attacker (distingiusher D^{F_k}) **does not** have access to the key k

Meaningless to distinguish F_k from f for a known key

- Recall: $F_k(x)$ is efficiently computable for any k, x
- D queries the oracle on x and gets a result y
- As D knows k (and x), it computes $y' = F_k(x)$
- If $y' = y$ output $\mathbf{1}$; else $\mathbf{0}$
- $\implies$ able to distinguish with $\mathbf{Pr} \approx \mathbf{1}$

PRF vs. RF

Warning

Attacker (distingiusher $\boldsymbol{D^{F_k}}$) **does not** have access to the key $\boldsymbol{k}$

Meaningless to distinguish $\boldsymbol{F_k}$ from $\boldsymbol{f}$ for a known key

- Recall: $\boldsymbol{F_k(x)}$ is efficiently computable for any $\boldsymbol{k, x}$
- $\boldsymbol{D}$ queries the oracle on $\boldsymbol{x}$ and gets a result $\boldsymbol{y}$
- As $\boldsymbol{D}$ knows $\boldsymbol{k}$ (and $\boldsymbol{x}$), it computes $\boldsymbol{y' = F_k(x)}$
- If $\boldsymbol{y' = y}$ output $\mathbf{1}$; else $\mathbf{0}$
- $\implies$ able to distinguish with $\mathbf{Pr} \approx \mathbf{1}$

Note

$\boldsymbol{f \in \mathcal{F}_n}$: $\mathbf{Pr}[\boldsymbol{f(x) = y'}] = \frac{\mathbf{1}}{\mathbf{2^n}}$

Is the Following PRF Secure?

$$\boldsymbol{F_k(x) = 0^n}$$

Is the Following PRF Secure?

$\boldsymbol{F_k(x) = 0^n}$

Distinguisher $\boldsymbol{D}$

1. Query $\boldsymbol{\mathcal{O}}$ on artbitrary $\boldsymbol{x}$: $\boldsymbol{y = \mathcal{O}(x)}$ (note: $\boldsymbol{\mathcal{O} = \{f, F_k\}}$)
2. If $\boldsymbol{y = 0^n}$ output $\mathbf{1}$; otherwise output $\mathbf{0}$

Is the Following PRF Secure?

$\boldsymbol{F_k(x) = 0^n}$

Distinguisher $\boldsymbol{D}$

1. Query $\boldsymbol{\mathcal{O}}$ on artbitrary $\boldsymbol{x}$: $\boldsymbol{y = \mathcal{O}(x)}$ (note: $\boldsymbol{\mathcal{O} = \{f, F_k\}}$)
2. If $\boldsymbol{y = 0^n}$ output $\mathbf{1}$; otherwise output $\mathbf{0}$

Analysis

$$\begin{aligned}
&|\mathbf{Pr}_{k \leftarrow \{0,1\}^n}[D^{F_k(\cdot)} = 1] - \mathbf{Pr}_{f \leftarrow \mathcal{F}_n}[D^{f(\cdot)} = 1]| \\
&= |1 - \frac{1}{2^n}| \approx 1 \not\leq \mathbf{negl}
\end{aligned}$$

Is the Following PRF Secure?

$$\boldsymbol{F_k(x) = k}$$

Is the Following PRF Secure?

$$F_k(x) = k$$

Distinguisher D

1. Query $\mathcal{O}$ on artbitrary x_1, x_2: $y_1 = \mathcal{O}(x_1), y_2 = \mathcal{O}(x_2)$
2. If $y_1 = y_2$ output 1; otherwise output 0

Is the Following PRF Secure?

$F_k(x) = k$

Distinguisher D

1. Query $\mathcal{O}$ on artbitrary x_1, x_2: $y_1 = \mathcal{O}(x_1), y_2 = \mathcal{O}(x_2)$
2. If $y_1 = y_2$ output $\mathbf{1}$; otherwise output $\mathbf{0}$

Analysis

$$|\Pr_{k \leftarrow \{0,1\}^n}[D^{F_k(\cdot)} = 1] - \Pr_{f \leftarrow \mathcal{F}_n}[D^{f(\cdot)} = 1]|$$
$$= |1 - \frac{1}{2^n}| \approx 1 \not\leq \mathbf{negl}$$

Is the Following PRF Secure?

$$\boldsymbol{F_k(x) = k \oplus x}$$

Is the Following PRF Secure?

$F_k(x) = k \oplus x$

Distinguisher D

1. Query $\mathcal{O}$ on artbitrary x_1, x_2: $y_1 = \mathcal{O}(x_1), y_2 = \mathcal{O}(x_2)$
2. If $(x_1 \oplus x_2) = (y_1 \oplus y_2)$ output $\mathbf{1}$; otherwise output $\mathbf{0}$

Is the Following PRF Secure?

$F_k(x) = k \oplus x$

Distinguisher D

1. Query $\mathcal{O}$ on artbitrary x_1, x_2: $y_1 = \mathcal{O}(x_1), y_2 = \mathcal{O}(x_2)$
2. If $(x_1 \oplus x_2) = (y_1 \oplus y_2)$ output $\mathbf{1}$; otherwise output $\mathbf{0}$

Analysis

$$|\mathbf{Pr}_{k \leftarrow \{0,1\}^n}[D^{F_k(\cdot)} = 1] - \mathbf{Pr}_{f \leftarrow \mathcal{F}_n}[D^{f(\cdot)} = 1]|$$
$$= |1 - \frac{1}{2^n}| \approx 1 \not\leq \mathbf{negl}$$

Is the Following PRF Secure?

$\mathcal{O} = F_k$

$$\mathbf{Pr}[x_1 \oplus x_2 = f(x_1) \oplus f(x_2)] = 1$$

$\mathcal{O} = f$

$$\mathbf{Pr}[x_1 \oplus x_2 = f(x_1) \oplus f(x_2)] =$$
$$\mathbf{Pr}[f(x_2) = x_1 \oplus x_2 \oplus f(x_1)] = \frac{1}{2^n}$$

PRFs vs. PRGs

PRF implies PRG

PRF F immediately implies PRG G:

- Define $G(k) = F_k(0 \ldots 0) | F_k(0 \ldots 1)$
- i.e. $G(k) = F_k(0_n) | F_k(1_n) | F_k(2_n) | \ldots$
  where i_n denotes the n-bit encoding of i
- Try to prove it formally (exercise 3.14).

PRF is a PRG with random access

PRF can be viewed as a PRG with random access to exponentially long output:

- The function F_k can be viewed as the $n2^n$-bit string
  $F_k(0 \ldots 0) | \ldots | F_k(1 \ldots 1)$

Permutations

Permutation

- Let $\boldsymbol{f} \in \boldsymbol{\mathcal{F}_n}$
- $\boldsymbol{f}$ is a *permutation* if it is a bijection
  - This means that the inverse $\boldsymbol{f^{-1}}$ exists
- Let $\boldsymbol{\mathcal{P}_n} \subset \boldsymbol{\mathcal{F}_n}$ be the set of permutations
- What is $|\boldsymbol{\mathcal{P}_n}|$?

$$|\boldsymbol{\mathcal{P}_n}| = \boldsymbol{2^n!}$$

Keyed Permutations

Keyed Permutation

- Let $\boldsymbol{F}$ be a length-preserving, keyed function
- $\boldsymbol{F}$ is a **keyed permutation** if
  1. $\boldsymbol{F_k}$ is a permutation for every $\boldsymbol{k}$ and
  2. $\boldsymbol{F_k^{-1}}$, the inverse of $\boldsymbol{F_k}$, is **efficiently computable**

Pseudorandom Permutations (PRPs)

Pseudorandom Permutation

- $\boldsymbol{F}$ is a **pseudorandom permutation** if $\boldsymbol{F_k}$, for uniform key $\boldsymbol{k} \in \{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{n}}$, is indistinguishable from a uniform permutation $\boldsymbol{f} \in \boldsymbol{\mathcal{P}_n}$
- Even if attacker can query the function **and its inverse**

PRP is Indistinguishable from PRF

Fact

A **random permutation is indistinguishable from a random function** for large enough $\boldsymbol{n}$

$\Longrightarrow$ in practice, PRPs are also good PRFs

Do PRFs/PRPs exist?

- PRF is a stronger primitive than PRG
  - PRF $\implies$ PRG
- We don't know if PRGs exist
- $\implies$ we don't know if PRFs exist

In practise

- Stream ciphers $\implies$ PRGs
- Block ciphers $\implies$ PRPs/PRFs

Next lecture

CPA-secure encryption using PRF/PRP

End

Reference: Section 3.5.1